R codes for the paper *Covariates impacts in spatial autoregressive models for compositional data*

C. Thomas-Agnan, T. Laurent and A. Ruiz-Gazen

Last update: 2023-01-12

This document presents the **R** codes used to obtain the computational results included in the paper "Covariates impacts in spatial autoregressive models for compositional data". To cite this work, please use:

C. Thomas-Agnan, T. Laurent and A. Ruiz-Gazen (2023). Covariates impacts in spatial autoregressive models for compositional data, *WP*.

Packages needed:

```
library(cartography) # mapping
library(compositions) # compositions data
library(gridExtra) # several frame in ggplot
library(Matrix)    # sparse matrix
library(scatterpie) # plot several pie
library(sp) # spatial package sp
library(spdep) # spatial econometrics
library(sf) # spatial package sf
library(tidyverse) # tidyverse universe
```

Section "Impacts decomposition and summaries" in the article

Create the data

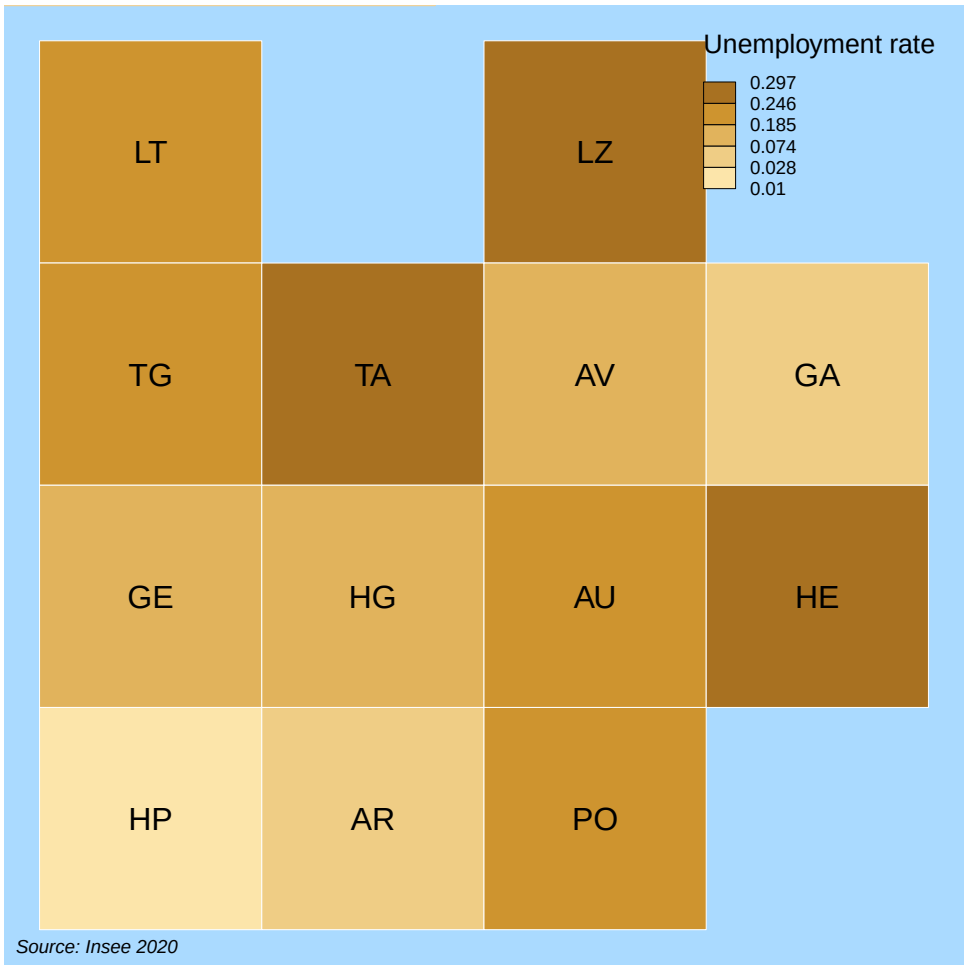
We consider the $n = 13$ departments of the Occitanie region in France whose geometries have been simplified. We simulate the unemployment rate using an uniform law with parameters $[0, 0.3]$.

```
n_occ <- 13
id_region_occ <- c("LT", "LZ", "TG", "TA", "AV", "GA",
                   "GE", "HG", "AU", "HE", "HP", "AR", "PO")
set.seed(12345)
occ_df <- data.frame(id = id_region_occ,
                     unemp = runif(n_occ, 0, 0.3),
                     row.names = "id")

sp_occ <- SpatialPointsDataFrame(cbind(c(0, 0, 0, 0, 1, 1, 1, 2, 2, 2, 2, 3, 3),
                                       c(0, 1, 2, 3, 0, 1, 2, 0, 1, 2, 3, 1, 2)),
                                 occ_df)

create_grid <- function(sp_centroid) {
  # initialization
  # define the grid
  coords <- coordinates(sp_centroid)
  id <- row.names(sp_centroid)
```

```
  nx <- length(unique(coords[, 1]))
  ny <- length(unique(coords[, 2]))
  gt <- GridTopology(c(min(coords[, 1]), min(coords[, 2])), c(1, 1), c(nx, ny))
  grd <- SpatialGrid(gt)
  spix <- as(grd, "SpatialPixels")
  spol <- as(spix, "SpatialPolygons")
  o <- over(spol, sp_centroid)[, 1]
  simu_spdf <- spol[!is.na(o), ]
  row.names(simu_spdf) <- id
  simu_spdf <- SpatialPolygonsDataFrame(simu_spdf, sp_centroid@data)
  return(simu_spdf)
}

spdf_occ <- create_grid(sp_centroid = sp_occ)
sf_occ <- as(spdf_occ, "sf")
```

Plotting the data

We represent the map of the "simulated" unemployment rate (not presented in the article):

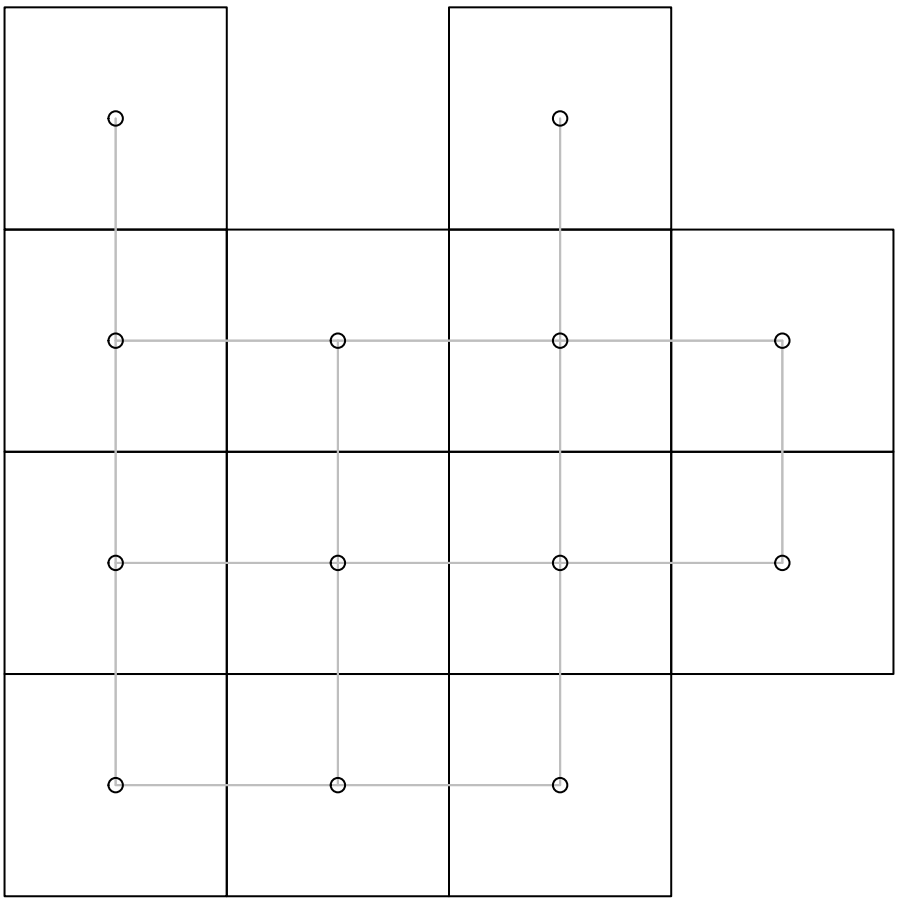


Initialization

Before simulating the spatial CoDa model:

- we define a spatial weight matrix based on the rook-contiguity (the plot was not presented in the article)

```
my_nb <- poly2nb(spdf_occ, queen = F)
W <- listw2mat(nb2listw(my_nb))
# pdf(file = "./figures/w.pdf", width = 8, height = 8)
par(oma = c(0, 0, 0, 0), mar = c(0, 0, 0, 0))
plot(spdf_occ)
plot(my_nb, coordinates(spdf_occ), col = "grey", add = T)
```



```
# dev.off()
```

Simulation

We simulate a $n \times 2$ matrix $\mathbf{Y}^*$ of the form (the choice of the coefficients have been inspired by the results of the spatial multivariate regression model found in Nguyen *et al.*, 2020)

$$\mathbf{Y}^* = \boldsymbol{\mu}^* + unemp \times \boldsymbol{\beta}^* + \mathbf{W}\mathbf{Y}^*\mathbf{R}^* + \boldsymbol{\epsilon}^*.$$

with $\boldsymbol{\mu}^* = (0.0696, -0.312)$, $\boldsymbol{\beta}^* = (-0.88, 1.82)$, $\mathbf{R}^* = \begin{bmatrix} 0.65 & 0 \\ 0.18 & 0.63 \end{bmatrix}$ and $\boldsymbol{\Sigma}^* = \begin{bmatrix} 0.09 & 0.02 \\ 0.02 & 0.09 \end{bmatrix}$.

We source the function that allows to simulate a spatial multivariate process (the function is described in http://author/code/spatial_coda/):

```
source(url("http://author/code/spatial_coda/R/simu_spatial_multi_y.R"))
```

```
my_x <- cbind(1, as.matrix(occ_df[, "unemp"]))
set.seed(12345)
y_star <- simu_spatial_multi_y(X = my_x,
                               beta_true = matrix(c(0.0696, -0.88,
                                                    -0.312, 1.82),
                                                  byrow = F, ncol = 2),
                               Sigma = matrix(c(0.09, 0.02, 0.02, 0.09), 2, 2),
                               RHO = matrix(c(0.65, 0, 0.18, 0.63), 2, 2),
                               W = W)
```

We transform the Y^* matrix back to the simplex for representing the three components:

```
Y <- ilrInv(y_star)
sf_occ[, c("y1", "y2", "y3")] <- Y
```

We represent the map of the different components of Y. The function *my_rgb_coda()* allows to choose appropriate colors:

```
source(url("http://www.thibault.laurent.free.fr/code/sp_coda_impacts/R/my_rgb_coda.R"))
```

```
source(url("http://author/code/sp_coda_impacts/R/my_rgb_coda.R"))
```

We present here some other alternatives for mapping compositional data. It can be particularly interesting if $D > 3$. First alternative is the waffle layer, second is the ternary diagram.

```
sf_occ$y1_100 <- round(sf_occ$y1 * 100)
sf_occ$y2_100 <- round(sf_occ$y2 * 100)
sf_occ$y3_100 <- round(sf_occ$y3 * 100)

sf_occ$y2_100[10] <- sf_occ$y2_100[10] - 1
sf_occ$y1_100[12] <- sf_occ$y1_100[12] + 1

R <- c(0, 0)
G <- c(1, 0)
B <- c(0.5, sqrt(3) / 2)
# create a grid of shares for Y
my_don_x_new <- Y[, 1] * R[1] + Y[, 2] * G[1] + Y[, 3] * B[1]
my_don_y_new  <- Y[, 1] * R[2] + Y[, 2] * G[2] + Y[, 3] * B[2]
```

To get Figure 1 in the article :

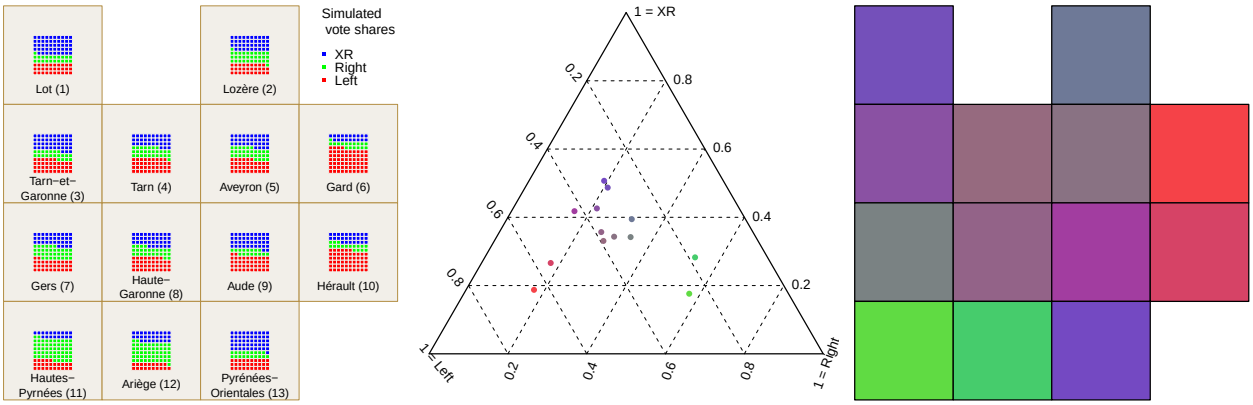
```
# pdf(file = "./figures/ternary_to_ilr.pdf", width = 10.5, height = 3.5)
par(oma = c(0, 0, 0, 0), mar = c(0, 0, 0, 0), mfrow = c(1, 3))
plot(st_geometry(sf_occ), col = "#f2efe9", border = "#b38e43", lwd = 0.5)
waffleLayer(
  x = sf_occ,
  var = c("y1_100", "y2_100", "y3_100"),
  cellvalue = 1,
  cellsize = 0.04,
  cellrnd = "ceiling",
  celltxt = "",
  labels = c("Left", "Right", "XR"),
  ncols = 10,
  col = my_rgb_coda(as(rbind(c(1, 0, 0), c(0, 1, 0), c(0, 0, 1)), "matrix"),
                    max_intensity = 1.5),
  border = "#f2efe9",
  legend.pos = "topright",
```

```
  legend.title.cex = 1,
  legend.title.txt = "Simulated \n vote shares",
  legend.values.cex = 1,
  add = TRUE
)
coords_dep <- st_coordinates(st_centroid(sf_occ))
```

```
## Warning in st_centroid.sf(sf_occ): st_centroid assumes attributes are constant
## over geometries of x
```

```
text(coords_dep[, 1], coords_dep[, 2]- 0.35,
     paste0(c("Lot", "Lozère", "Tarn-et-\nGaronne", "Tarn", "Aveyron",
       "Gard", "Gers", "Haute-\nGaronne", "Aude", "Hérault",
       "Hautes-\nPyrnées", "Ariège", "Pyrénées-\nOrientales"),
       paste0(" (", 1:13, ")")), cex = 0.9
)

plot(rbind(R, G, B),
     xaxt = "n",
     yaxt = "n",
     frame = F,
     pch = 16,
     type = "n",
     xlab = "",
     ylab = "",
     asp = 1,
     ylim = c(-0.1, sqrt(3)/2)
)

lines(c(0, 1), c(0, 0), col = "black")
lines(c(0, 0.5), c(0, sqrt(3)/2), col = "black")
lines(c(1, 0.5), c(0, sqrt(3)/2), col = "black")

base_trian <- (G[1] - R[1])
hauteur <- B[2] - R[2]

# XR lines
for (k in 1:4) {
  lines(c(R[1] + base_trian * k/10, G[1] - base_trian * k/10),
        c(R[2] + k/5 * hauteur, R[2] + k/5 * hauteur), lty = 2, lwd = 0.8)
  text(G[1] - base_trian * k/10, R[2] + k/5 * hauteur, 2 * k/10, pos = 4, cex = 1)
}
text(G[1] - base_trian * 1/2, R[2] + hauteur, "1 = XR", pos = 4, cex = 1)

# Left lines
for (k in 1:4) {
  lines(c(R[1] + k/5, base_trian / 2 + k * 1/10),
        c(R[2], B[2] - k/5 * hauteur), lty = 2, lwd = 0.8)
  text(R[1] + k/5 * base_trian, R[2] - 0.02, 2 * k/10, pos = 1, cex = 1, srt = 70)
}
text(R[1] + base_trian, R[2], "1 = Right", pos = 1, cex = 1, srt = 70)

# Right lines
for (k in 1:4) {
```

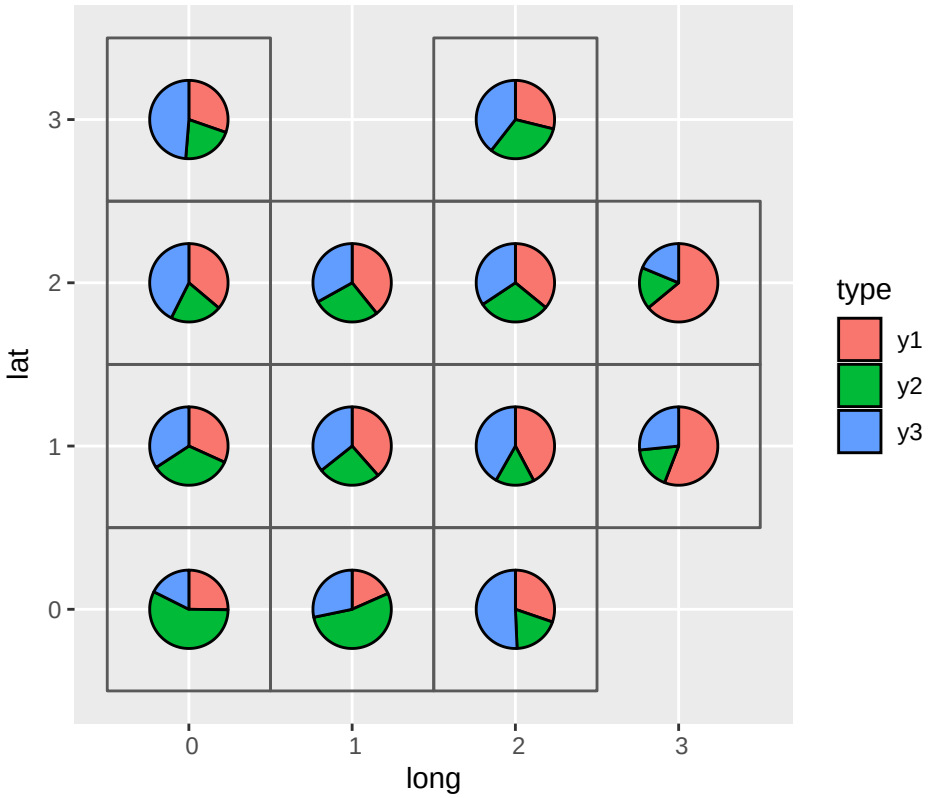
```
  lines(c(R[1] + base_trian * k * 1 / 10, R[1] + base_trian * k/5),
        c(R[2] + k/5 * hauteur, R[2]), lty = 2, lwd = 0.8)
  text(R[1] + base_trian * k * 1 / 10, R[2] + k/5 * hauteur, 1 - 2 * k/10,
       pos = 2, cex = 1, srt = -50)
}
text(R[1]-0.04,  R[2] + 0.02, "1 = Left",
       pos = 4, cex = 1, srt = -50)

points(my_don_x_new, my_don_y_new, pch = 16,
       col = my_rgb_coda(as(Y, "matrix"), max_intensity = 1.5),
       cex = 1)
#text(c(0.04, 0.95, 1/2), c(0-0.06, -0.06, sqrt(3)/2 - 0.005),
#     c("Left", "Right", "XR"), pos = c(2, 4, 3),
#     cex = 0.7)
# map
plot(st_geometry(sf_occ), col = my_rgb_coda(as(Y, "matrix"), max_intensity = 1.5))
```




```
# dev.off()
```

Second alternative is based on the pie (figure not presented in the article):

```
sf_occ[ , c("y1", "y2", "y3")] <- as(Y, "matrix")
sf_occ[ , c("long", "lat")] <- st_coordinates(st_centroid(sf_occ))
sf_occ$region <- row.names(sf_occ)
ggplot()  +
  geom_scatterpie(aes(x = long, y = lat, group = region), data = st_drop_geometry(sf_occ),
                  cols = c("y1", "y2", "y3"), pie_scale = 4) +
  coord_equal() +
  geom_sf(data = sf_occ, fill = NA)
```



Summary statistics (not presented in the article)

```
kableExtra::kbl(round(t(sapply(
  st_drop_geometry(sf_occ)[ , c("unemp", "y1", "y2", "y3")],
  function(x) c(min = min(x), max = max(x),
                mean = mean(x), median = median(x),
                sd = sd(x)))), 3))
```

```
## Warning in !is.null(rmarkdown::metadata$output) && rmarkdown::metadata$output
## %in% : 'length(x) = 2 > 1' dans la conversion automatique vers 'logical(1)'
```

	min	max	mean	median	sd
unemp	0.010	0.297	0.169	0.216	0.094
y1	0.184	0.640	0.367	0.359	0.122
y2	0.160	0.572	0.286	0.258	0.132
y3	0.176	0.507	0.347	0.343	0.102

Computing the semi-elasticities

The function *sp_semi_elast_Ycompo()* computes the semi-elasticities individual by individual in the spatial-compositional regression model. It takes as input arguments:

- **X**, the covariates matrix of size $n \times p$ (with the intercept if included in the model)
- **b_star** the matrix of the estimated parameter $\hat{\beta}^*$ of size $p \times (D-1)$, in the spatial model in the ILR space,
- **index_b**, an integer, the column index of the variable that will be considered to compute the semi-elasticities
- **W**, the spatial weight matrix of the n observations
- **RHO** the matrix of the estimated spatial autocorrelation parameter
- **type**, a character with **"exact"** or **"appro"** that indicated the method to be used

- **print_summary**, a logical, if TRUE, print the summary

```
source(url("http://author/code/sp_coda_impacts/R/sp_semi_elast_Ycompo.R"))
```

Application: we use function *sp_semi_elast_Ycompo()* to compute the semi-elasticities on our simulated data:

```
my_beta <- matrix(c(0.0696, -0.88, -0.312, 1.82),
                  byrow = F, ncol = 2)
elast_sp <- sp_semi_elast_Ycompo(my_x, my_beta, index_b = 2,
                                 method = "appro", W = W,
                                 RHO = matrix(c(0.65, 0, 0.18, 0.63), 2, 2))
```

```
## Direct impact:
##  -0.3992917 -1.465008 1.755953
##  Indirect impact:
##  -1.571031 -0.4834733 2.309066
##  Total impact:
##  -1.970323 -1.948481 4.065019
```

Representing the semi-elasticities

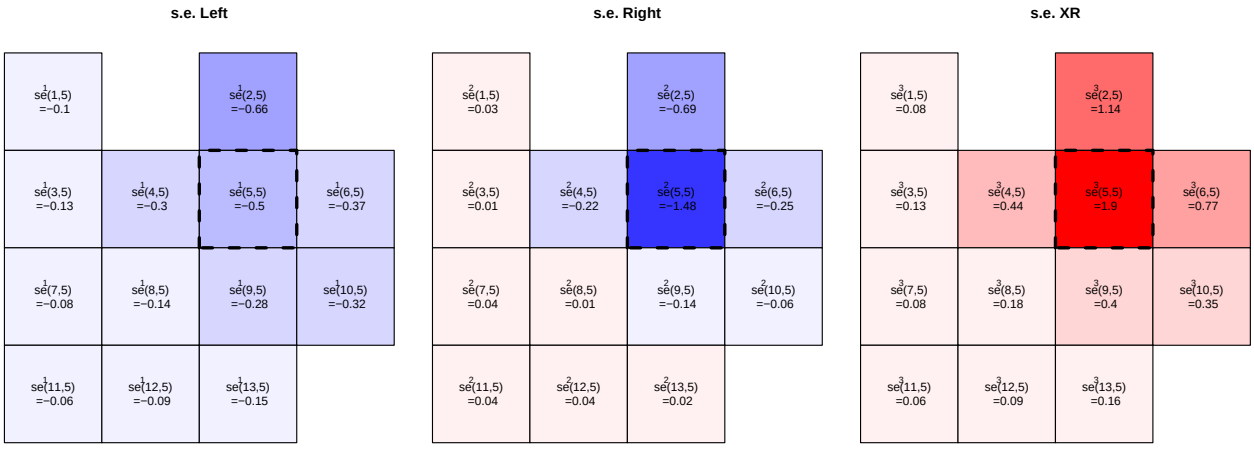
We can map the matrices of semi-elasticities

```
rwb <- colorRampPalette(colors = c("blue", "white", "red"))
bks <- seq(-max(abs(range(as.vector(elast_sp)))),
           max(abs(range(as.vector(elast_sp)))), length.out = 21)
```

For instance we map the impacts due to observation 5 on all the network:

```
k <- 5
#pdf("figures/impacts_s5.pdf", width = 10.5, height = 3.5)
op <- par(mfrow = c(1, 3), oma = c(0, 0.5, 1, 0), mar = c(0, 0.5, 1, 0),
          mgp = c(1, 1, 1))
plot(st_geometry(sf_occ), col = rwb(20)[findInterval(elast_sp[, k, 1],
     bks, all.inside = T)], main = "s.e. Left",
     ylab = paste0("Impacts due to change of x at s", k))

plot(st_geometry(sf_occ[5, ]), add = T, lwd = 3, lty = 2)

text(st_coordinates(st_centroid(sf_occ))[, 1], st_coordinates(st_centroid(sf_occ))[, 2],
      paste0(paste("se(", 1:13,",", 5, ")", sep = ""), "\n =",
             round(elast_sp[, k, 1], 2)))
text(st_coordinates(st_centroid(sf_occ))[, 1] - 0.08,
     st_coordinates(st_centroid(sf_occ))[, 2]+ 0.04, "1",
     cex = 0.7, pos = 3)
plot(st_geometry(sf_occ), col = rwb(20)[findInterval(elast_sp[, k, 2],
                           bks, all.inside = T)],
     main = "s.e. Right")
text(st_coordinates(st_centroid(sf_occ))[, 1], st_coordinates(st_centroid(sf_occ))[, 2],
      paste0(paste("se(", 1:13,",", 5, ")", sep = ""), "\n =",
             round(elast_sp[, k, 2], 2)))
text(st_coordinates(st_centroid(sf_occ))[, 1] - 0.1,
     st_coordinates(st_centroid(sf_occ))[, 2]+ 0.04, "2",
     cex = 0.7, pos = 3)
plot(st_geometry(sf_occ[5, ]), add = T, lwd = 3, lty = 2)
```

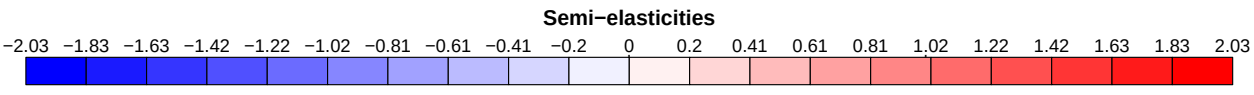
```
plot(st_geometry(sf_occ), col = rwb(20)[findInterval(elast_sp[, k, 3],
                                        bks, all.inside = T)],
     main = "s.e. XR")
text(st_coordinates(st_centroid(sf_occ))[, 1], st_coordinates(st_centroid(sf_occ))[, 2],
      paste0(paste("se(", 1:13,",", 5, ")", sep = ""), "\n =",
             round(elast_sp[, k, 3], 2)))
text(st_coordinates(st_centroid(sf_occ))[, 1] - 0.08,
     st_coordinates(st_centroid(sf_occ))[, 2]+ 0.04, "3",
     cex = 0.7, pos = 3)
plot(st_geometry(sf_occ[5, ]), add = T, lwd = 3, lty = 2)
```




```
par(op)
#dev.off()
```

With the follwing color codes:

```
#pdf("figures/se_colors.pdf", width = 12, height = 1.5)
par(oma = c(0, 0, 0, 0), mar = c(0, 0, 0, 0))
plot(c(0, 1), c(0, 0), frame.plot = F, xaxt = "n",
     yaxt = "n", ylab = "", pch = "|",
     xlab = "", xlim = c(-2.05, 2.05))
title("Semi-elasticities", line = -2)
for(k in 1:(length(bks) - 1)) {
  polygon(c(bks[k], bks[k+1], bks[k+1], bks[k], bks[k]),
          c(-0.28, -0.28, 0.08, 0.08, -0.28),
          col = (rwb(20))[k])
  text(bks[k], 0, round(bks[k], 2), pos = 3)
}
  text(bks[k+1], 0, round(bks[k+1], 2), pos = 3)
```

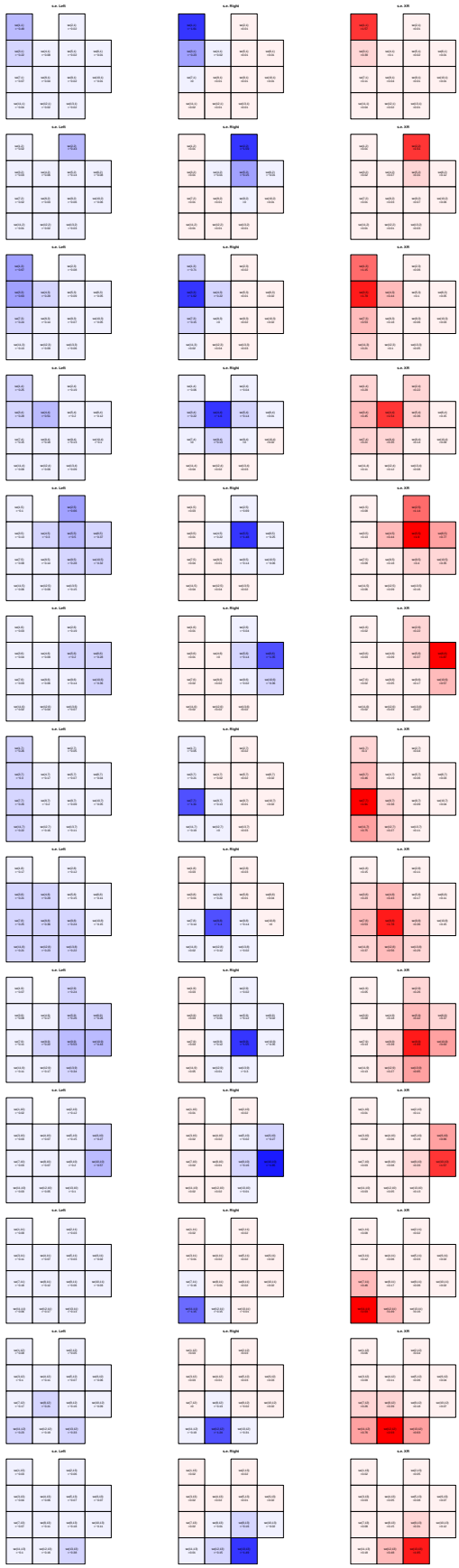



```
#dev.off()
```

We can map the impacts to each observation (not presented in the article) :

```
op <- par(mfrow = c(13, 3), oma = c(0, 2, 2, 0), mar = c(0, 2, 2, 0),
          mgp = c(1, 1, 1))
for(k in 1:13) {
 plot(st_geometry(sf_occ), col = rwb(20)[findInterval(elast_sp[, k, 1],
```

```
                                          bks, all.inside = T)],
    main = "s.e. Left", ylab = paste0("Impacts due to change of x at s", k))
 text(st_coordinates(st_centroid(sf_occ))[, 1], st_coordinates(st_centroid(sf_occ))[, 2],
      paste0(paste("se(", 1:13, ",", k, ")", sep = ""), "\n =",
             round(elast_sp[, k, 1], 2)))
 plot(st_geometry(sf_occ), col = rwb(20)[findInterval(elast_sp[k, , 2],
                                        bks, all.inside = T)],
    main = "s.e. Right")
 text(st_coordinates(st_centroid(sf_occ))[, 1], st_coordinates(st_centroid(sf_occ))[, 2],
      paste0(paste("se(", 1:13, ",", k, ")", sep = ""), "\n =",
             round(elast_sp[, k, 2], 2)))
 plot(st_geometry(sf_occ), col = rwb(20)[findInterval(elast_sp[, k, 3],
                                            bks, all.inside = T)],
    main = "s.e. XR")
 text(st_coordinates(st_centroid(sf_occ))[, 1], st_coordinates(st_centroid(sf_occ))[, 2],
      paste0(paste("se(", 1:13, ",", k, ")", sep = ""), "\n =",
             round(elast_sp[, k, 3], 2)))
}
```

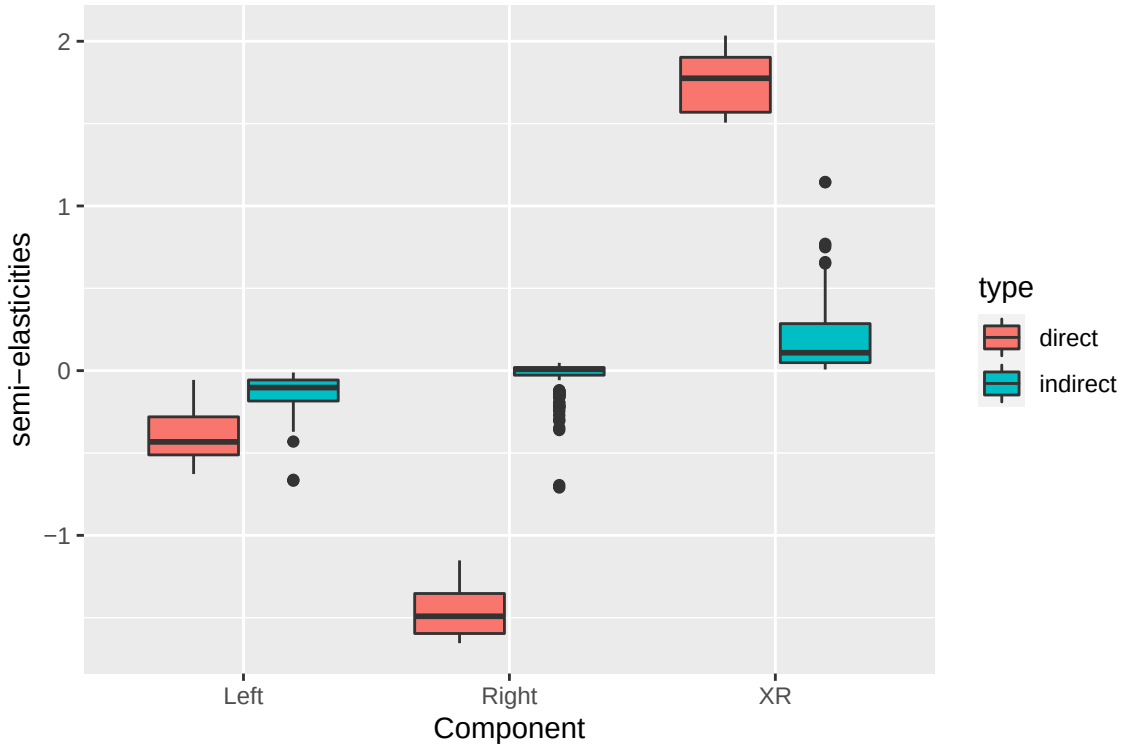

s.e. Left
s.e. Right
s.e. 3/5
Impedance bridge in a circ.
Wave law in charged rod
Impedance bridge in a di
Capacitance diagram in a
Inside law in charge in a di
Capacitance in a point
Wave law in charged rod
Impedance bridge in a point
Capacitance diagram in a
Inside law in charge in a di
Impedance bridge in a circ.
Wave law in charged rod
Inside law in charge in a di

```
par(op)
```

We now plot the boxplot of the elasticities with respect to their group: direct, indirect

```
D <- 3
res_boxplot <- data.frame(
  obs = rep(rep(sf_occ$region, each = n_occ), times = D),
  elas = c(as.vector(elast_sp[ , , 1]),
           as.vector(elast_sp[ , , 2]),
           as.vector(elast_sp[ , , 3])),
  type = ifelse(as.vector(diag(n_occ)) == 1, "direct", "indirect"),
  comp = rep(c("Left", "Right", "XR"), each = n_occ ^ 2))
```

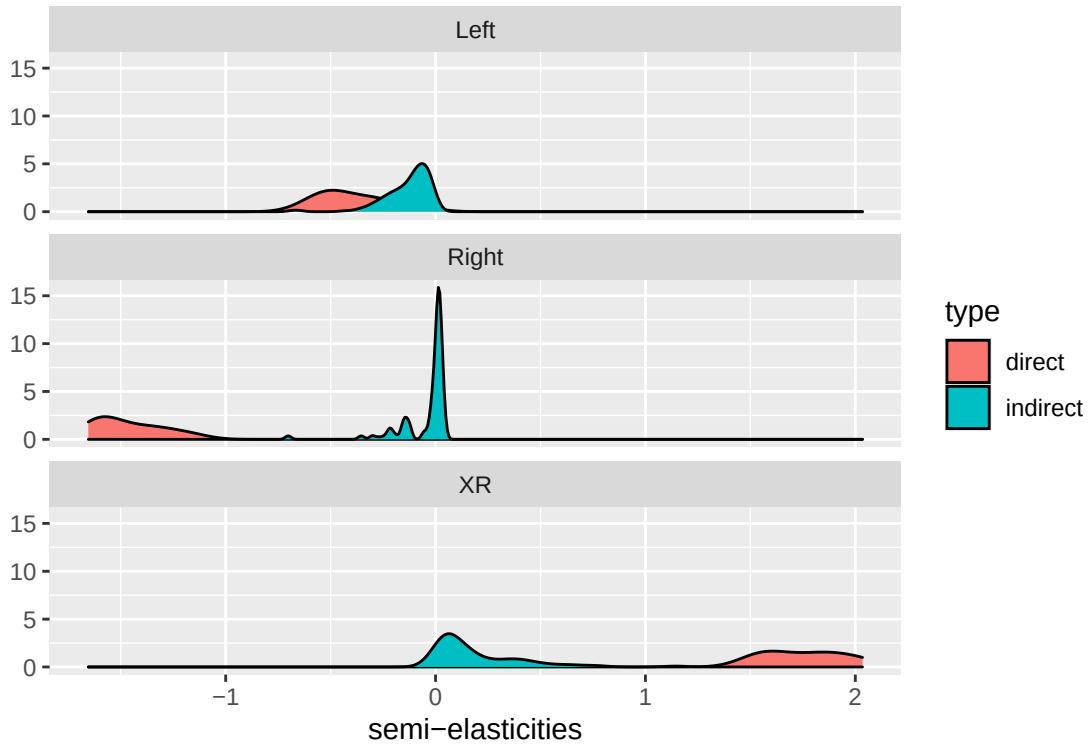
```
ggplot(data = res_boxplot) +
  aes(y = elas, fill = type, x = comp) +
  geom_boxplot() +
  xlab("Component") +
  ylab("semi-elasticities")
```




```
#ggsave("figures/boxplot_semi.pdf", width = 6, height = 4)
```

We plot the density plot of the semi-elasticities:

```
ggplot(data = res_boxplot) +
  aes(x = elas, fill = type) +
  geom_density() +
  facet_wrap(~comp, ncol = 1) +
  xlab("semi-elasticities") +
  ylab("")
```




```
# ggsave("figures/density_semi.pdf", width = 5, height = 4)
```

Groupping the semi-elasticities

The function *map_elas_ternary()* takes as argument :

- **df_coords** the coordinates of the spatial observations
- **X**, the covariates matrix of size $n \times p$ (with the intercept if included in the model)
- **b_star** the matrix of the estimated parameter $\hat{\beta}^*$ in the spatial model in the ILR coordinates, of size $p \times (D-1)$
- **index_b**, an integer, the column index of the variable that will be considered to compute the semi-elasticities
- **W**, the spatial weight matrix of the n observations
- **RHO** the matrix of the estimated spatial autocorrelation parameter
- **delta**, the value of the increment to compute the $\hat{Y}(x+\delta)$
- **type**, a integer equals to 1 or 2. The first case focuses on the impacts aggregated by spatial unit whereas the second one illustrates the impacts due a spatial unit on all others
- **scale_ternary**, a logical, if TRUE, plot the scale on the ternary diagram
- **add**, a logical, if TRUE, do not open a new graphic device
- **range**, corresponds to $x_{max} - x_{min}$ where x_{min} is the minimum of the x-geographical coordinate and x_{max} is the maximum of the x-geographical coordinate
- **cex.pts**, the size of the points
- **size**, a scalar that fits the size of the ternary diagram
- **length**, the size of the arrow
- **cex.legend**, the size of the police
- **legend.ternary**, a vector of size 3 of characters that give the legend for the ternary diagram.
- **ternary_unit**, a logical, if TRUE prints the scaling of the ternary diagram

```
source(url("http://author/code/sp_coda_impacts/R/map_elas_ternary.R"))
```

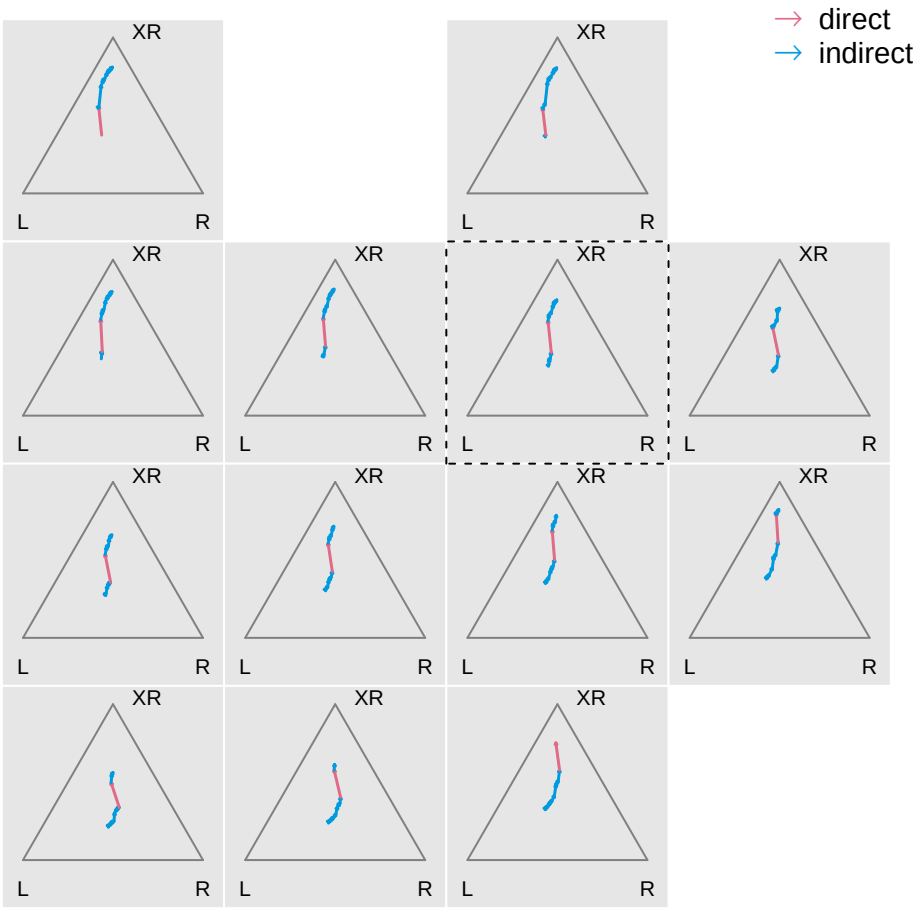
```r
coords_s <- st_coordinates(st_centroid(sf_occ))
rownames(coords_s) <- sf_occ$region
```

Alternative 1

We propose to make a zoom on the direct $DE(i)$ and indirect $IE(i)$ observed at each location i.

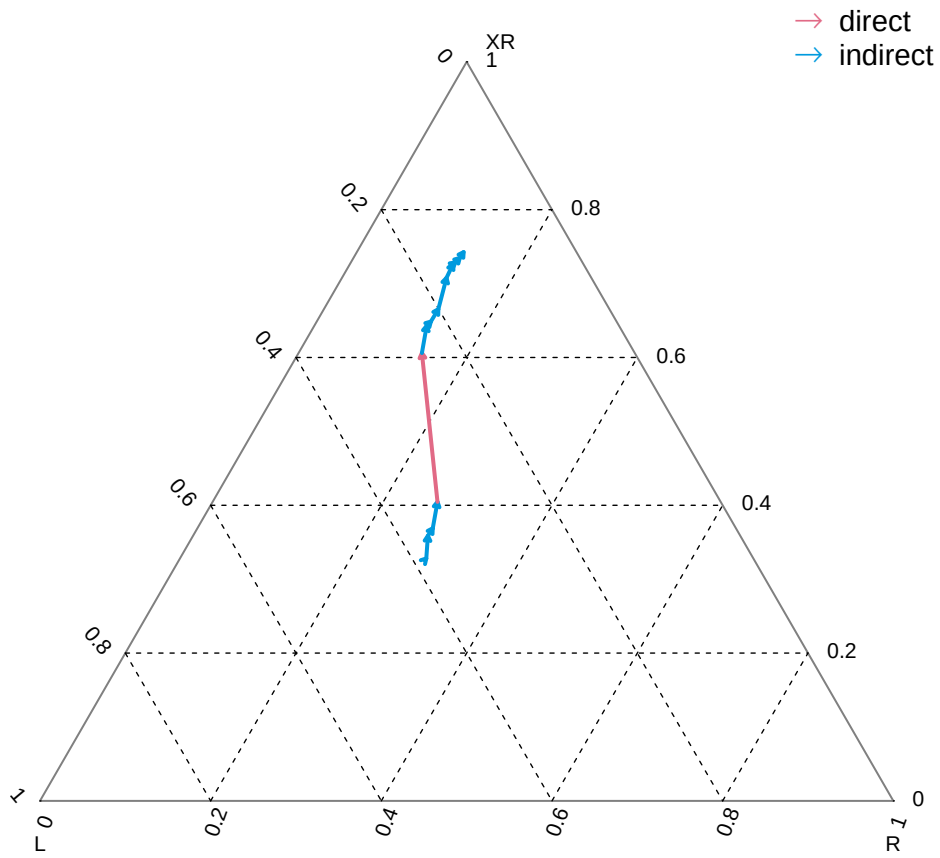
In red, we represent the total impact, in red the direct impact and in green the indirect impact.

```r
#pdf("figures/local_v2.pdf", width = 6, height = 6)
par(oma = c(0, 0, 0, 0), mar = c(0, 0, 0, 0))
plot(st_geometry(sf_occ), col = rgb(0.9, 0.9, 0.9), border = "white")
coords_s_ts <- coords_s
coords_s_ts[, 2] <- coords_s_ts[, 2] - 0.05
map_elas_ternary(coords_s_ts,
                 my_x, my_beta, index_b = 2, W = W,
                 RHO = matrix(c(0.65, 0, 0.18, 0.63), 2, 2), delta = 0.3,
                 range = 3, size = 2.7, lwd = 1.5,  length = 0.01, cex.legend = 0.65,
                            legend.ternary = c("L", "R", "XR"))
plot(st_geometry(sf_occ[5, ]), add = T, lwd = 1, lty = 2)

legend("topright", legend = c("direct", "indirect"), pch = c(NA, NA),
       lwd = 3, col = c("#E16A86", "#009ADE"),
       cex = 0.9, lty = NA, box.lwd = -1) #normal legend. Do not plot line
par(font = 5) #change font to get arrows
legend("topright", legend = c("        ", "          "), pch = c(174, 174),
       lwd = 3, col = c("#E16A86", "#009ADE"),
       cex = 0.9, lty = NA, bty = "n", box.lwd = -1)
```




```
#dev.off()
```

```
#pdf("figures/zoom_aveyron_2.pdf", width = 5, height = 5)
par(oma = c(0, 0, 0, 0), mar = c(0, 0, 0, 0))
plot(st_geometry(sf_occ[5, ]), border = "white", ylim = c(1.7, 2.5))
map_elas_ternary(coords_s, add = T, row_to_plot = 5,
                  X = my_x, b_star = my_beta, index_b = 2, W = W,
                 type = 1, scale_ternary = TRUE,
                  RHO = matrix(c(0.65, 0, 0.18, 0.63), 2, 2),
                 delta = 0.3,
                  range = 3, cex.pts = 0.2, size = 3.2,
                 length = 0.03,
                 cex.legend = 0.7, lwd = 2,
                 legend.ternary = c("L", "R", "XR"))
legend("topright", legend = c("direct", "indirect"), pch = c(NA, NA),
       lwd = 3, col = c("#E16A86", "#009ADE"),
       cex = 0.9, lty = NA, box.lwd = -1) #normal legend. Do not plot line
par(font = 5) #change font to get arrows
legend("topright", legend = c("        ", "          "), pch = c(174, 174),
       lwd = 3, col = c("#E16A86", "#009ADE"),
       cex = 0.9, lty = NA, bty = "n", box.lwd = -1)
```

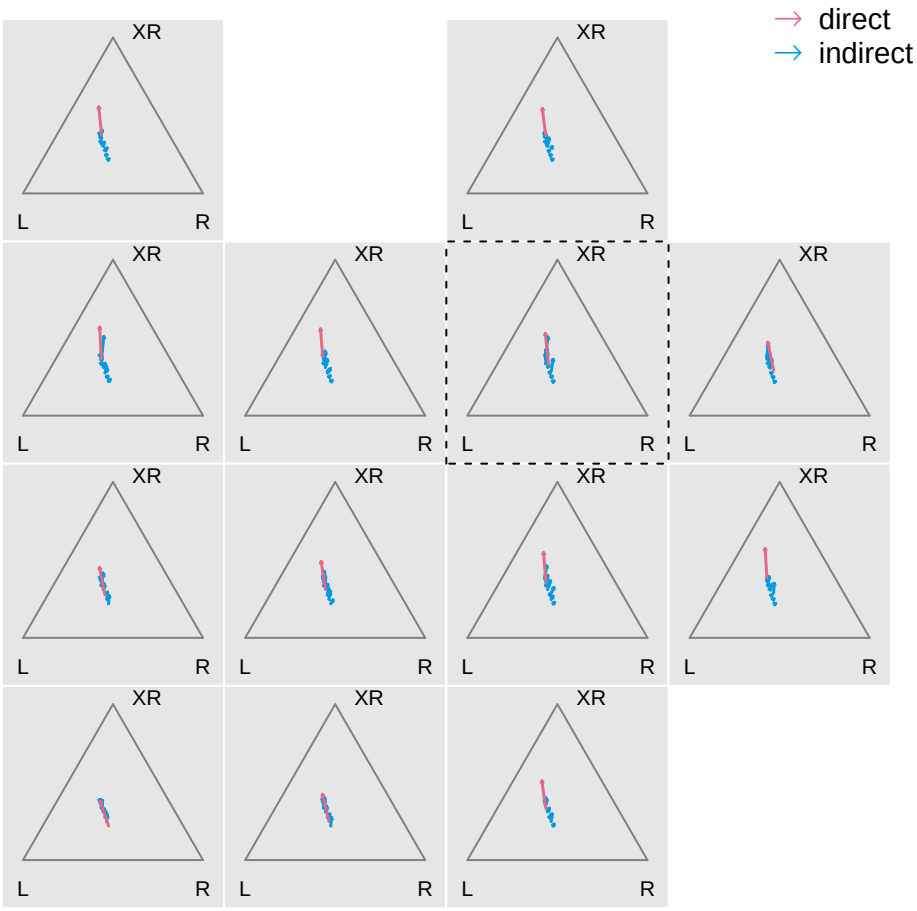



```
#dev.off()
```

Alternative 2

We aim to plot the impacts caused by changing X at one particular observation.

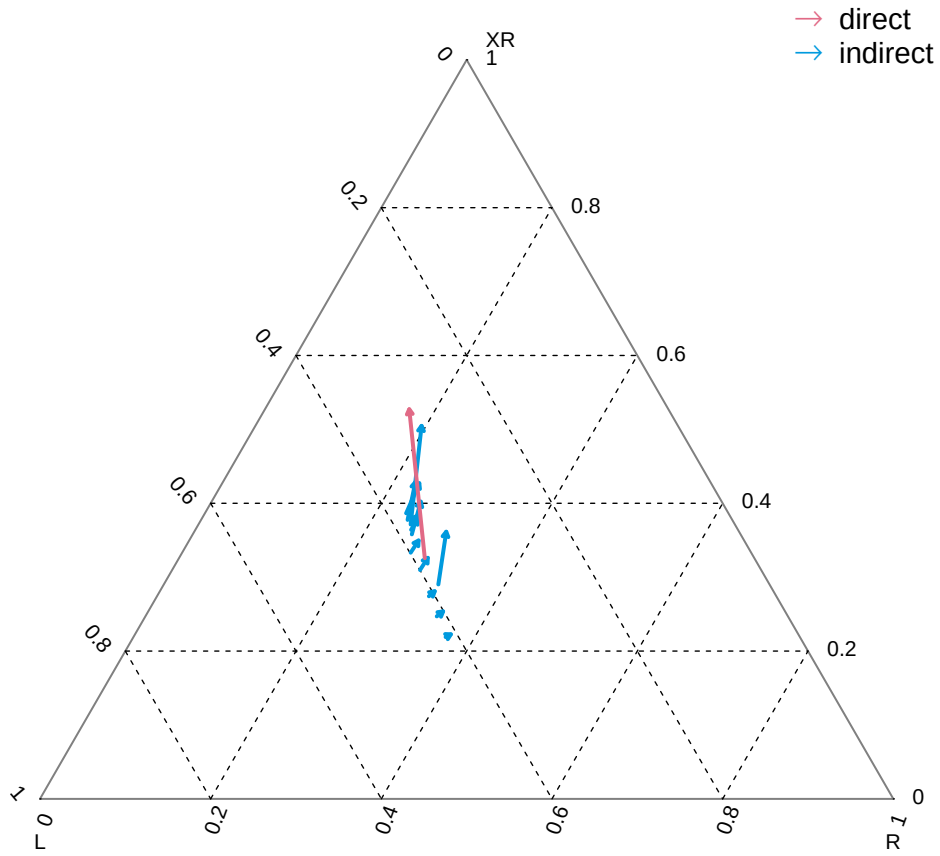
```
#pdf("figures/local_v1.pdf", width = 6, height = 6)
par(oma = c(0, 0, 0, 0), mar = c(0, 0, 0, 0))
coords_s_ts <- coords_s
coords_s_ts[, 2] <- coords_s_ts[, 2] - 0.05
plot(st_geometry(sf_occ), col = rgb(0.9, 0.9, 0.9), border = "white")
map_elas_ternary(df_coords = coords_s_ts,
                  X = my_x, b_star = my_beta, index_b = 2, W = W,
                 type = 2,
                  RHO = matrix(c(0.65, 0, 0.18, 0.63), 2, 2), delta = 0.3,
                  range = 3, cex.pts = 0.2, size = 2.7, lwd = 1.5,  length = 0.01,
                 cex.legend = 0.65, legend.ternary = c("L", "R", "XR"))
plot(st_geometry(sf_occ[5, ]), add = T, lwd = 1, lty = 2)
legend("topright", legend = c("direct", "indirect"), pch = c(NA, NA),
       lwd = 3, col = c("#E16A86", "#009ADE"),
       cex = 0.9, lty = NA, box.lwd = -1) #normal legend. Do not plot line
par(font = 5) #change font to get arrows
legend("topright", legend = c("        ", "          "), pch = c(174, 174),
       lwd = 3, col = c("#E16A86", "#009ADE"),
       cex = 0.9, lty = NA, bty = "n", box.lwd = -1)
```




```
#dev.off()
```

We zoom on the Aveyron departement:

```
#pdf("figures/zoom_aveyron_1.pdf", width = 5, height = 5)
par(oma = c(0, 0, 0, 0), mar = c(0, 0, 0, 0))
plot(st_geometry(sf_occ[5, ]), border = "white", ylim = c(1.7, 2.5))
map_elas_ternary(coords_s, add = T, row_to_plot = 5,
                  X = my_x, b_star = my_beta, index_b = 2, W = W,
                 type = 2, scale_ternary = TRUE,
                  RHO = matrix(c(0.65, 0, 0.18, 0.63), 2, 2), delta = 0.3,
                  range = 3, cex.pts = 0.2, size = 3.2,
                 length = 0.03,
                 cex.legend = 0.7, lwd = 2,
                 legend.ternary = c("L", "R", "XR"))
legend("topright", legend = c("direct", "indirect"), pch = c(NA, NA),
       lwd = 3, col = c("#E16A86", "#009ADE"),
       cex = 0.9, lty = NA, box.lwd = -1) #normal legend. Do not plot line
par(font = 5) #change font to get arrows
legend("topright", legend = c("        ", "          "), pch = c(174, 174),
       lwd = 3, col = c("#E16A86", "#009ADE"),
       cex = 0.9, lty = NA, bty = "n", box.lwd = -1)
```




```
#dev.off()
```

Aggregating the semi-elasticities into local direct and indirect impacts

We summarize the results into direct, indirect and total effects:

```
spatial_direct <- apply(elast_sp, 3, function(x) diag(x))
spatial_total <- apply(elast_sp, c(1, 3), sum)
spatial_indirect <- spatial_total - spatial_direct
colnames(spatial_direct) <- colnames(spatial_total) <-
  colnames(spatial_indirect) <- c("Left", "Right", "XR")
```

Barplot of the semi-elasticities

We can map the three impacts $DE(i), IE(i), TE(i)$ at each location i on the same map. We have implemented function *map_elas()* that allows to print the local effects. The imput arguments are :

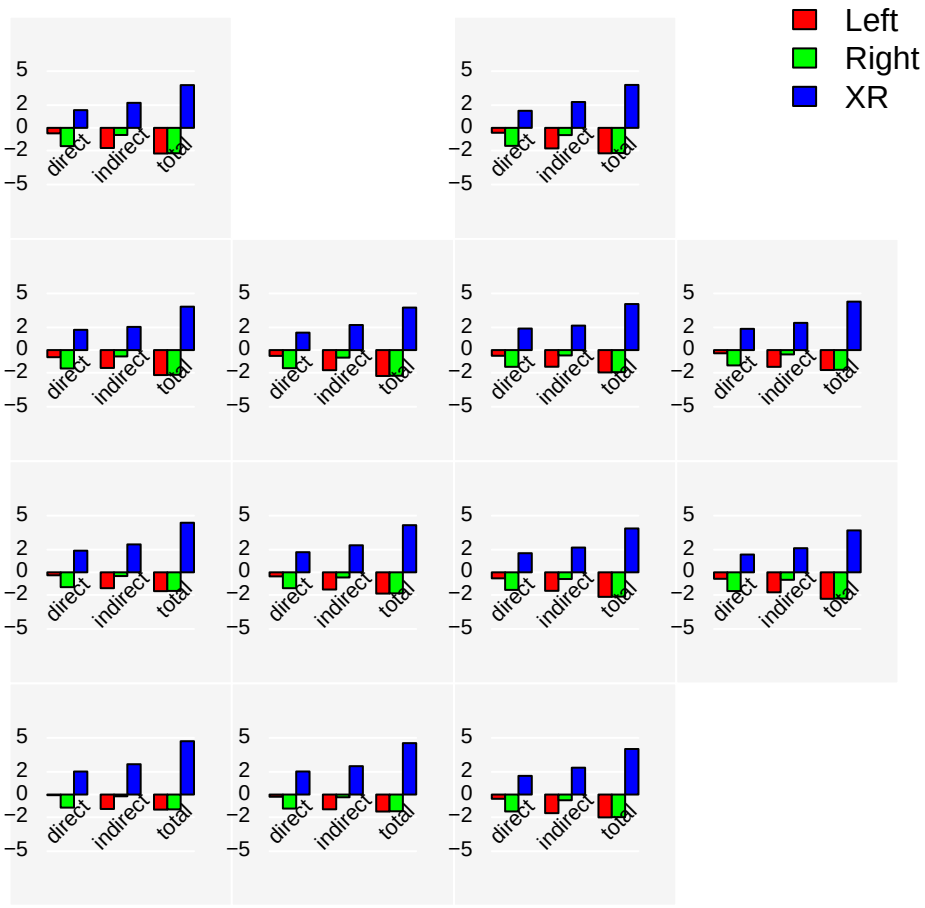
- **spatial_direct**, **spatial_indirect**, **spatial_total**, the matrices of size $n \times D$ of the direct, indirect and total impacts
- **coords_s**, the matrices of size $n \times 2$ of the spatial coordinates. **row.names** must be given to print the legend
- **q4**, the vector of colors for each component $d \in D$. The ordering is the same than the columns of **spatial_direct**
- **add**, a logical indicating whether a new graphic window may be opened.
- **max_bar** the value of the highest bar of the bar plots
- **max_width** the value of the width of the bars
- **x.legend**, a character among "none", "bottomright", "bottom", "bottomleft", "left", "topleft", "top", "topright", "right", "center" that gives the location of the legend

- **label_s**, a logical for printing the names of the entities
- **plot_scale**, a logical, if TRUE prints the scaling

```
source(url("http://author/code/sp_coda_impacts/R/map_elas_bar.R"))
```

```
coords_s <- st_coordinates(st_centroid(sf_occ))
rownames(coords_s) <- sf_occ$region
```

```
#pdf("figures/bar_local.pdf", width = 5, height = 5)
par(oma = c(0, 0, 0, 0), mar = c(0, 0, 0, 0))
plot(st_geometry(sf_occ),
     col = rgb(0.96, 0.96, 0.96), border = "white")
map_elas_bar(spatial_direct, spatial_indirect, spatial_total, coords_s, add = T,
          q4 = my_rgb_coda(as(rbind(c(1, 0, 0), c(0, 1, 0), c(0, 0, 1)), "matrix"),
                    max_intensity = 1.5),
         width_bar = 4, max_bar = 4, cex.legend = 0.65, x.legend = "topright",
         label_s = F, round.scale = 0)
```



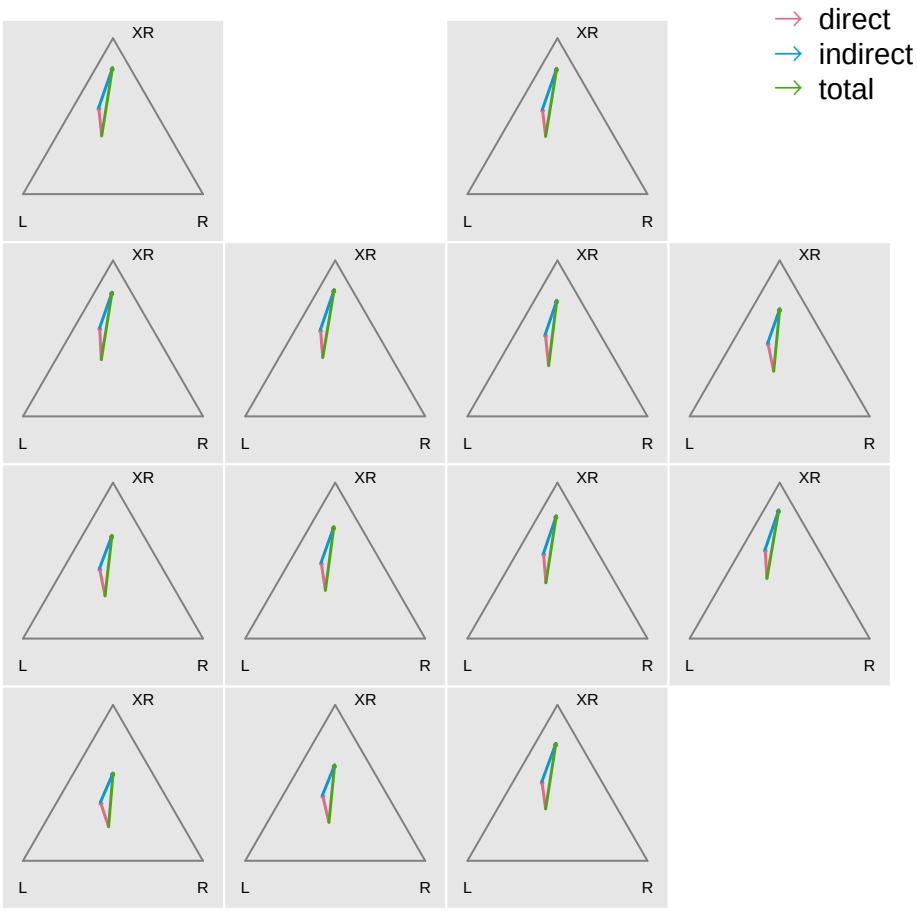

```
#dev.off()
```

Ternary diagram of the local impacts

We aggregate the indirect impacts and plot them in the ternary diagram

```
#pdf("figures/local_v3.pdf", width = 5, height = 5)
par(oma = c(0, 0, 0, 0), mar = c(0, 0, 0, 0))
plot(st_geometry(sf_occ), col = rgb(0.9, 0.9, 0.9), border = "white")
```

```
map_elas_ternary(coords_s_ts,
                 my_x, my_beta, index_b = 2, W = W, all_impacts = F,
                 RHO = matrix(c(0.65, 0, 0.18, 0.63), 2, 2), delta = 0.3,
                 range = 3, size = 2.7, lwd = 1.5,  length = 0.01, cex.legend = 0.5,
                            legend.ternary = c("L", "R", "XR"))

#legend("topright", legend = c("direct", "indirect", "total"), lty = 1,
#        col = c("#E16A86", "#009ADE", "#50A315"), cex = 0.9, box.lwd = -1)

legend("topright", legend = c("direct", "indirect", "total"), pch = c(NA, NA, NA),
       lwd = 3, col = c("#E16A86", "#009ADE", "#50A315"),
       cex = 0.9, lty = NA, box.lwd = -1) #normal legend. Do not plot line
par(font = 5) #change font to get arrows
legend("topright", legend = c("        ", "          ", "       "), pch = c(174, 174, 174),
       lwd = 3, col = c("#E16A86", "#009ADE", "#50A315"),
       cex = 0.9, lty = NA, bty = "n", box.lwd = -1)
```




```
#dev.off()
```

Case study

Preparation of the data

We first import the data:

```
load(url("http://author/code/spatial_coda/R/data_cantons.RData"))
source(url("http://author/code/spatial_coda/R/preparation_base.R"))
```

Preparation of the data :

```
Ye <- as(y_ilr, "matrix")
Xe <- as(cbind(1, x2_df[, c("diplome3_ilr1", "diplome3_ilr2",
                            "employ_ilr1", "employ_ilr2", "employ_ilr3",
                            "employ_ilr4",
                            "age3_ilr1", "age3_ilr2",
                            "unemp_rate", "income_rate", "voters")]),
         "matrix")
ne <- nrow(Xe)
k <- ncol(Xe)
```

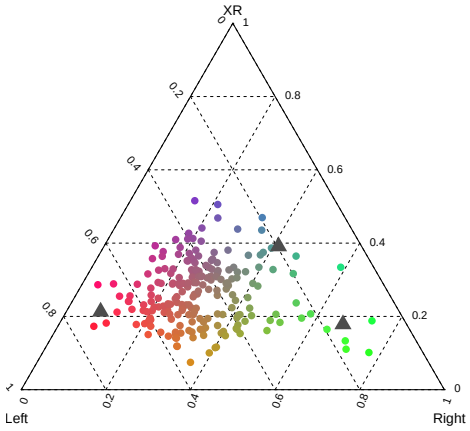
Spatial weight matrix :

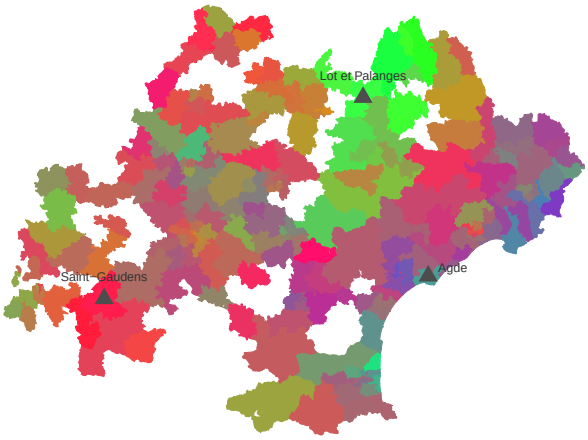
```
coords_fr <- st_coordinates(st_centroid(contours_occitanie_no0))
W_listw <- nb2listw(knn2nb(knearneigh(coords_fr[, 1:2], 5)),
                    style = "W")
W_dep <- listw2mat(W_listw)
```

We represent the vote shares:

```
R <- c(0, 0)
G <- c(1, 0)
B <- c(0.5, sqrt(3) / 2)
Y <- as(ilrInv(Ye), "matrix")
# create a grid of shares for Y
my_don_x_new <- Y[, 1] * R[1] + Y[, 2] * G[1] + Y[, 3] * B[1]
my_don_y_new  <- Y[, 1] * R[2] + Y[, 2] * G[2] + Y[, 3] * B[2]

# create a grid of shares for Y
#pdf(file = "figures/vote_shares_cantons.pdf", width = 11, height = 4)
op <- par(oma = c(1, 1, 0, 1), mar = c(0.7, 0.7, 0.1, 0.7),
          mfrow = c(1, 2))
plot(rbind(R, G, B),
     xaxt = "n",
     yaxt = "n",
     frame = F,
     pch = 16,
     type = "n",
     xlab = "",
     ylab = "",
     asp = 1,
     ylim = c(-0.1, sqrt(3)/2 +0.05)
)


lines(c(0, 1), c(0, 0), col = "black")
lines(c(0, 0.5), c(0, sqrt(3)/2), col = "black")
lines(c(1, 0.5), c(0, sqrt(3)/2), col = "black")

# XR lines
for (k in 0:5) {
  lines(c(k/10, 1-k/10), c(k/5 * B[2], k/5 * B[2]), lty = 2, lwd = 0.8)
```

```
  text(1-k/10, k/5 * B[2], 2 * k/10, pos = 4, cex = 0.7)
}

# Left lines
for (k in 0:5) {
  lines(c(k/5, 0.5 + k * 1/10), c(0, B[2] - k/5 * B[2]), lty = 2, lwd = 0.8)
  text(k/5, 0, 2 * k/10, pos = 1, cex = 0.7, srt = 70)
}

# Right lines
for (k in 0:5) {
  lines(c(k * 1 / 10, k/5), c(k/5 * B[2], 0), lty = 2, lwd = 0.8)
  text(k * 1 / 10, k/5 * B[2], 1 - 2 * k/10, pos = 2, cex = 0.7, srt = -50)
}

points(my_don_x_new, my_don_y_new, pch = 16,
       col = my_rgb_coda(Y, max_intensity = 1.5),
       cex = 1)
text(c(0.04, 0.95, 1/2), c(0 - 0.07, -0.07, sqrt(3)/2 - 0.005),
     c("Left", "Right", "XR"), pos = c(2, 4, 3),
     cex = 0.8)

points(my_don_x_new[c(72, 24, 94)], my_don_y_new[c(72, 24, 94)],
       pch = 17,
       col = rgb(0.3, 0.3, 0.3),
       cex = 1.7)

plot(st_geometry(contours_occitanie_no0),
     col = my_rgb_coda(Y, max_intensity = 1.5),
     border = my_rgb_coda(Y, max_intensity = 1.5), lwd = 0.1)
plot(st_geometry(st_centroid(contours_occitanie_no0[c(72, 24, 94), ])),
       add = T, pch = 17,
       col = rgb(0.3, 0.3, 0.3), cex = 1.9)
coords_pts <- st_coordinates(st_geometry(st_centroid(
  contours_occitanie_no0[c(72, 24, 94), ])))
text(coords_pts[, 1], coords_pts[, 2] + 5000,
     contours_occitanie_no0[c(72, 24, 94), ]$NOMCT,
     pos = c(3, 3, 4), cex = 0.78, col = rgb(0.2, 0.2, 0.2))
```





```
par(op)
#dev.off()
```

Spatial CoDa Model :

```
source(url("http://author/code/spatial_coda/R/estimate_spatial_multi_gen_N.R"))
```

```
W_listw <- nb2listw(knn2nb(knearneigh(coords_fr[, 1:2], 5)),
                    style = "W")
W_dep <- listw2mat(W_listw)
(res_sar_N <- estimate_spatial_multi_gen_N(Y = Ye, X = Xe, W = W_dep,
                  method = "s2sls",
                  ind_RHO = matrix(c(T, T, T, T), 2, 2), compute_sd = T)
)
```

```
## $est_beta
##              [,1]        [,2]
##  [1,] -3.00976250 -2.43174165
##  [2,] -0.72443013 -0.45860592
##  [3,]  0.07294022 -0.57604703
##  [4,] -0.11236449 -0.11670102
##  [5,]  0.34774036  0.03935649
##  [6,] -0.18650684  0.07886590
##  [7,]  0.12997343 -0.02449034
##  [8,] -0.90095105  0.31593358
##  [9,]  0.65800437 -0.63820565
## [10,] -0.82822463  1.73470257
## [11,]  3.39571871  0.64503901
## [12,]  0.07393745  0.14599409
##
## $est_RHO
##              [,1]      [,2]
## [1,] 0.6539724925 0.1762922
## [2,] 0.0008498897 0.6355886
##
## $est_GAMMA
##      [,1] [,2]
## [1,]    0    0
## [2,]    0    0
##
## $est_SIGMA
##             [,1]        [,2]
## [1,]  0.15190419 -0.02762565
## [2,] -0.02762565  0.04080749
##
## $sd_beta
##             [,1]       [,2]
##  [1,] 1.15727962 0.59982333
##  [2,] 0.46147965 0.23918702
##  [3,] 0.53097311 0.27520580
##  [4,] 0.12078677 0.06260434
##  [5,] 0.14630758 0.07583189
##  [6,] 0.09882756 0.05122278
```

```
##  [7,] 0.05400310 0.02799005
##  [8,] 0.40224773 0.20848684
##  [9,] 0.32027444 0.16599971
## [10,] 2.85615235 1.48035685
## [11,] 0.81334917 0.42156260
## [12,] 0.08150794 0.04224594
##
## $sd_RHO
##            [,1]       [,2]
## [1,] 0.15413648 0.17601965
## [2,] 0.07988964 0.09123179
##
## $sd_GAMMA
##      [,1] [,2]
## [1,]    0    0
## [2,]    0    0
```

Semi-elasticities:

```
my_beta <- res_sar_N$est_beta
elast_sp <- sp_semi_elast_Ycompo(Xe, my_beta, index_b = 11, method = "exact",
                                 W = W_dep,
                                 RHO = res_sar_N$est_RHO)
```

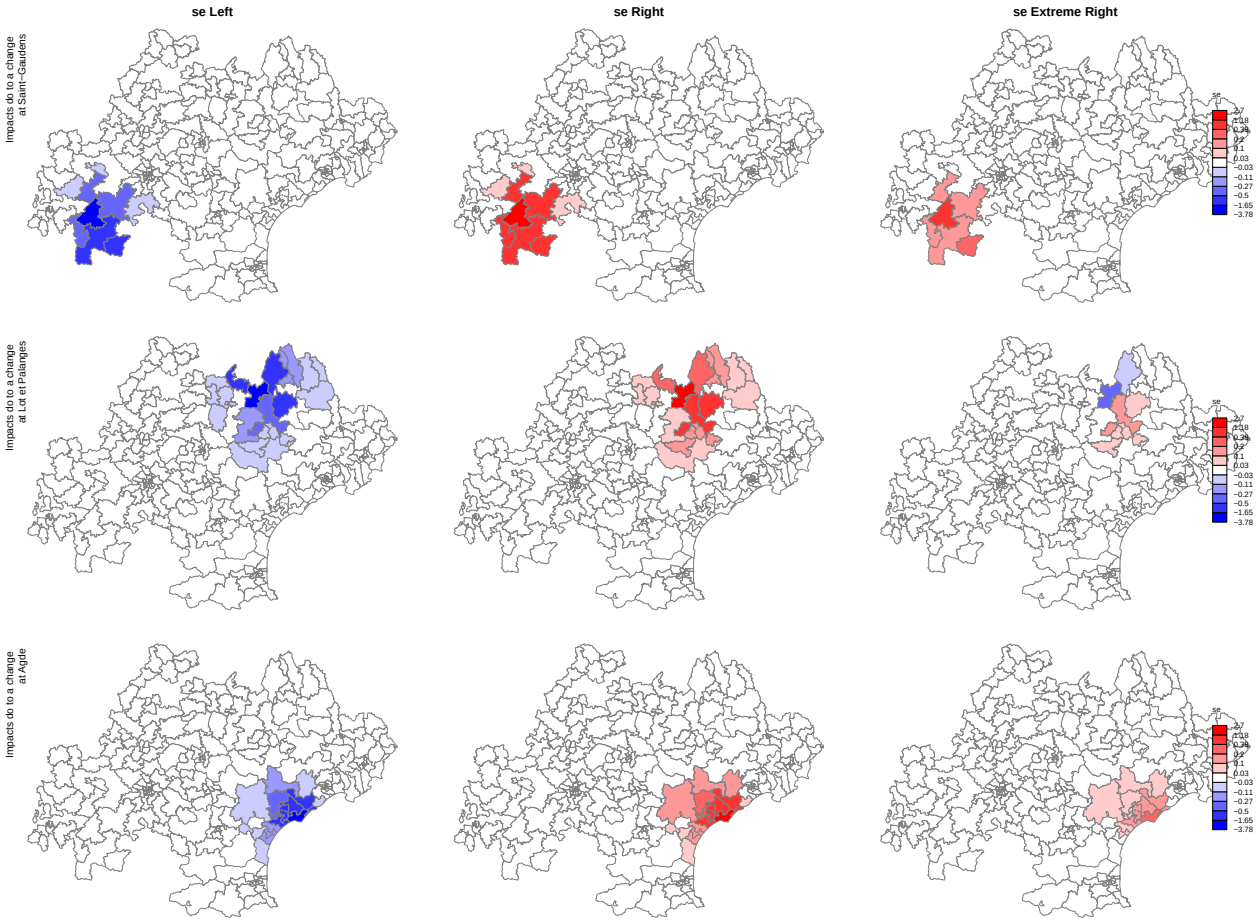
```
## Direct impact:
##  -2.515609 2.926981 1.080577
##  Indirect impact:
##  -4.428969 5.300119 1.759229
##  Total impact:
##  -6.944578 8.2271 2.839806
```

We represent the semi-elasticities in the three cantons chosen in the article (Saint-Gaudens, Lot et Palanges, Agde):

```
# fix the bounds of the colors
elas_k <- c(elast_sp[c(72, 24, 94), , 1],
            elast_sp[c(72, 24, 94), , 2],
            elast_sp[c(72, 24, 94), , 3])

pos_class <- classInt::classIntervals(elas_k[elas_k>0], 6, "kmeans")
neg_class <- classInt::classIntervals(elas_k[elas_k<0], 6, "kmeans")
# we plot the se on the map
rwb <- colorRampPalette(colors = c("blue", "white", "red"))
```

```
require(cartography)
# pdf("figures/elasticites_spatial_canton_123.pdf", width = 8.5, height = 5.5)
op <- par(mfrow = c(3, 3), oma = c(0, 1, 0, 0), mar = c(0, 0, 1, 0))
i <- 1
for(k in c(72, 24, 94)) {
  contours_occitanie_no0$elas_1 <- elast_sp[k, , 1]
  contours_occitanie_no0$elas_2 <- elast_sp[k, , 2]
  contours_occitanie_no0$elas_3 <- elast_sp[k, , 3]
  # we add one class for positive and negative
  choroLayer(contours_occitanie_no0, var = "elas_1",
             breaks = c(neg_class$brks[1:6], pos_class$brks[2:7]),
             col = rwb(11),legend.pos = "n", legend.values.rnd = 0,
```

```
            legend.title.txt = "se (left)", border = rgb(0.5, 0.5, 0.5), lwd = 0.2)
  if (i == 1) {
    title("se Left")
text(390000, 2000000, "Impacts do to a change \n at Saint-Gaudens",
     pos = 2, cex = 1, srt = 90)
  }
  if (i == 2) {
    text(390000, 2000000, "Impacts do to a change \n at Lot et Palanges",
     pos = 2, cex = 1, srt = 90)
  }
  if (i == 3) {
    text(390000, 2000000, "Impacts do to a change \n at Agde",
     pos = 2, cex = 1, srt = 90)
  }
  choroLayer(contours_occitanie_no0, var = "elas_2",
             breaks = c(neg_class$brks[1:6], pos_class$brks[2:7]),
             col = rwb(11),legend.pos = "n", legend.values.rnd = 0,
             legend.title.txt = "se (right)", border = rgb(0.5, 0.5, 0.5), lwd = 0.2)
  if (i == 1)
    title("se Right")
  choroLayer(contours_occitanie_no0, var = "elas_3",
             breaks = c(neg_class$brks[1:6], pos_class$brks[2:7]),
             col = rwb(11),legend.pos = c(760000, 1800000),
             legend.values.rnd = 2, legend.title.txt = "se",
             border = rgb(0.5, 0.5, 0.5), lwd = 0.2, legend.values.cex = 0.8)
  if (i == 1)
    title("se Extreme Right")
  i <- i + 1
}
```



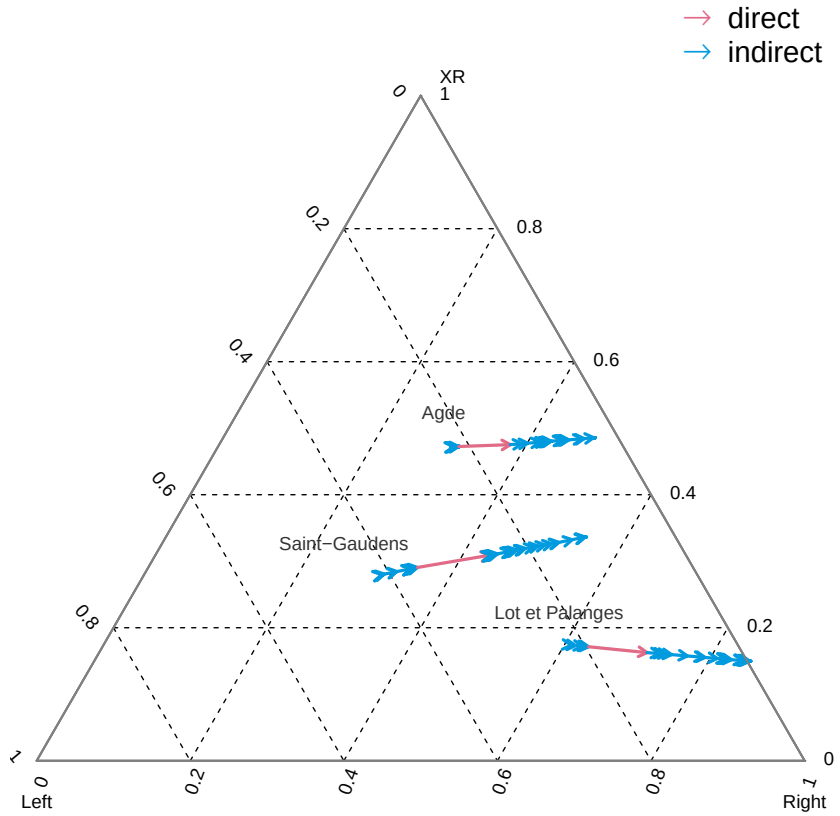

```
par(op)
# dev.off()
```

We zoom on the three cantons :

```
coords_s <- st_coordinates(st_centroid(contours_occitanie_no0))
```

```
# pdf("figures/zoom_canton_2.pdf", width = 5, height = 5)
par(oma = c(1, 1, 0, 1), mar = c(0.7, 0.7, 0.1, 0.7))
map_elas_ternary(coords_s, add = F, row_to_plot = 72,
                  X = Xe, b_star = my_beta, index_b = 11, W = W_dep,
                 type = 1, scale_ternary = TRUE,
                  RHO = matrix(c(0.65, 0, 0.18, 0.63), 2, 2), delta = 0.1,
                   ternary_unit = T, range = 1,
                 cex.pts = 0.2,  length = 0.05,
                 cex.legend = 0.6, lwd = 1.6,
                 legend.ternary = c("Left", "Right", "XR"))
text(0.4, 0.25, "Saint-Gaudens", cex = 0.6, col = rgb(0.2, 0.2, 0.2), pos = 3)
map_elas_ternary(coords_s, add = T, row_to_plot = 24,
                  X = Xe, b_star = my_beta, index_b = 11, W = W_dep,
                 type = 1, scale_ternary = FALSE,
                  RHO = matrix(c(0.65, 0, 0.18, 0.63), 2, 2), delta = 0.1,
                   ternary_unit = T, range = 1,
                 cex.pts = 0.2,  length = 0.05,
                 cex.legend = 0.6, lwd = 1.6,
                 legend.ternary = c("", "", ""))
```

```
text(0.53, 0.42, "Agde", cex = 0.6, col = rgb(0.2, 0.2, 0.2), pos = 3)
map_elas_ternary(coords_s, add = T, row_to_plot = 94,
                  X = Xe, b_star = my_beta, index_b = 11, W = W_dep,
                 type = 1, scale_ternary = FALSE,
                  RHO = matrix(c(0.65, 0, 0.18, 0.63), 2, 2), delta = 0.1,
                   ternary_unit = T, range = 1,
                 cex.pts = 0.2,  length = 0.05,
                 cex.legend = 0.6, lwd = 1.6,
                 legend.ternary = c("", "", ""))
text(0.68, 0.16, "Lot et Palanges", cex = 0.6, col = rgb(0.2, 0.2, 0.2), pos = 3)
legend("topright", legend = c("direct", "indirect"), pch = c(NA, NA),
       lwd = 3, col = c("#E16A86", "#009ADE"),
       cex = 0.9, lty = NA, box.lwd = -1) #normal legend. Do not plot line
par(font = 5) #change font to get arrows
legend("topright", legend = c("        ", "          "), pch = c(174, 174),
       lwd = 3, col = c("#E16A86", "#009ADE"),
       cex = 0.9, lty = NA, bty = "n", box.lwd = -1)
```



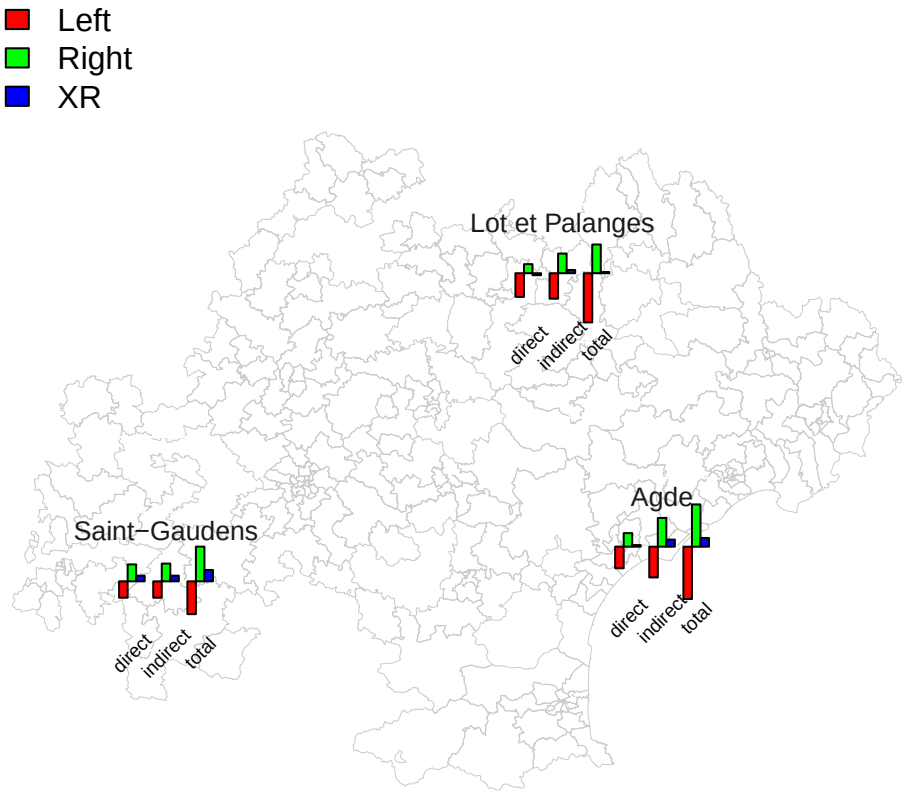

```
# dev.off()
```

We summarize the results into direct, indirect and total effects:

```
spatial_direct <- apply(elast_sp, 3, function(x) diag(x))
spatial_total <- apply(elast_sp, c(1, 3), sum)
spatial_indirect <- spatial_total - spatial_direct
colnames(spatial_direct) <- colnames(spatial_total) <-
  colnames(spatial_indirect) <- c("Left", "Right", "XR")
```

We now plot the individual barplot

```
# pdf("figures/bar_local_cantons.pdf", width = 6, height = 3.5)
par(oma = c(0, 0, 0, 0), mar = c(0, 0, 0, 0))
plot(st_geometry(contours_occitanie_no0),
     border = rgb(0.8, 0.8, 0.8), lwd = 0.1)
map_elas_bar(spatial_direct, spatial_indirect, spatial_total, coords_s, add = T,
             index_to_plot = c(72, 24, 94), plot_scale = F,
             q4 = my_rgb_coda(as(rbind(c(1, 0, 0), c(0, 1, 0), c(0, 0, 1)), "matrix"),
                    max_intensity = 1.5),
         width_bar = 2, max_bar = 12, cex.legend = 0.6, x.legend = "topleft",
         label_s = F, round.scale = 0)
text(coords_pts[, 1], coords_pts[, 2]+10000,
     contours_occitanie_no0[c(72, 24, 94), ]$NOMCT,
     pos = 3, cex = 0.8, col = rgb(0.1, 0.1, 0.1))
```




```
# dev.off()
```